AD-A100 914 YALE UNIV NEW HAVEN CT DEPT OF COMPUTER SCIENCE F/G 12/1
PRECONDITIONED CONJUGATE-GRADIENT METHODS FOR NONSYMMETRIC SYST--ETC(U)
APR 81 H C ELMAN N00014-76-C-0277
UNCLASSIFIED RR-203 NL

1 OF 1
AD A
100914

END
DATE
FILMED
8-81
DTIC

LEVEL

(12)

YALE UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

12

PRECONDITIONED CONJUGATE-GRADIENT METHODS FOR NONSYMMETRIC SYSTEMS OF LINEAR EQUATIONS

Howard C. Elman[1]

Research Report # 203

April 1981

[1]This research was supported by ONR Grant N00014-76-C-0277, and ONR Grant N00014-80-0076 under subcontract from Florida State University.

1. Introduction

In this paper, we present a class of *iterative descent* methods for solving large, sparse, nonsymmetric systems of linear equations whose coefficient matrices have positive-definite symmetric parts. Such problems commonly arise from the discretization of non-self-adjoint elliptic partial differential equations. The methods we consider are modelled after the conjugate gradient method (CG) [1, 4]. They require no estimation of parameters and their rate of convergence appears to depend on the spectrum of A rather than A^TA. Their convergence can also be accelerated by preconditioning techniques.

The methods are tested on two sample problems, and their numerical behavior is compared with that of two other methods, the nonsymmetric Chebyshev algorithm [10, 11], and the conjugate gradient method applied to the normal equations [8, 9, 13]. All the methods are tested in conjunction with two preconditionings, the incomplete LU factorization [12], and the modified incomplete LU factorization [3, 6].

In Section 2, we describe the methods, outline their computational costs, and present some bounds on their convergence rates. In Section 3, we show how they can be implemented with preconditioning techniques. In Section 4, we describe two sample nonsymmetric model problems derived from a non self adjoint elliptic equation. Finally, in Section 5, we present the results of numerical experiments with the methods. Tables and figures follow the list of references.

2. The Generalized Conjugate *Residual* Method, and Variants

In this section, we describe a class of descent methods for solving the linear system

$$A\,x = f \quad . \tag{2.1}$$

where A is a nonsymmetric matrix of order N with positive definite symmetric part. We consider *four* variants, all of which have the following form:

Choose x_0 .
Compute $r_0 = f - Ax_0$.
Set $p_0 = r_0$.
FOR $i = 0$ STEP 1 UNTIL Convergence DO

$$a_i = \frac{(r_i, Ap_i)}{(Ap_i, Ap_i)}$$

$$x_{i+1} = x_i + a_i p_i$$

$$r_{i+1} = r_i - a_i Ap_i$$

Compute p_{i+1} .

The choice of a_i minimizes $\|b - A(x_i + ap_i)\|_2 = \|r_{i+1}\|_2$ as a function of a, so that the Euclidean norm of the residual decreases at each step. The four methods are determined by four techniques for choosing p_{i+1}:

(1) Generalized Conjugate Residual (GCR):

$$p_{i+1} = r_{i+1} + \sum_{j=0}^{i} b_j^{(i)}\, p_j \quad . \tag{2.2}$$

where

$$b_j^{(i)} = -\frac{(Ar_{i+1}, Ap_j)}{(Ap_j, Ap_j)} \quad , \quad j \le i \quad . \tag{2.3}$$

(2) Orthomin(k) [18]:

$$p_{i+1} = r_{i+1} + \sum_{j=\max(0,i-k+1)}^{i} b_j^{(i)} p_j \quad .$$

where $\{b_j^{(i)}\}$ are defined by (2.3).

(3) GCR(k):

The generalized conjugate residual algorithm restarted every k+1 steps. Every k+1 steps, the current iterate, $x_{j(k+1)}$ is taken as the new starting guess.

(4) Minimum Residual (MR): $p_{i+1} = r_{i+1}$.

The direction vectors $\{p_i\}$ generated by GCR are constructed so that

$$(Ap_i, Ap_j) = 0 \text{ for } i \neq j \quad .$$

As a result, x_i minimizes the functional $E(w) = \|f-Aw\|_2$ over the affine space $x_0 + \text{span}\{p_0,\ldots,p_{i-1}\}$, and

$$\|r_i\|_2 = \min_{q_i \in P_i} \|q_i(A)r_0\|_2 \quad . \tag{2.4}$$

GCR is the analogue of the conjugate residual method (CR) [2, 15] for symmetric problems. If A is symmetric and positive-definite, then (2.2) reduces to a two-term expression and the resulting algorithm is equivalent to CR.

Orthomin(k) has been proposed as an alternative to GCR that is less expensive in terms of both work per iteration and storage. The vector p_{i+1} is orthogonal to only the last k (≥ 0) vectors $\{p_j\}_{j=i-k+1}^{i}$. Only k direction vectors need to be stored. The iterate x_i minimizes E(w) over the affine space $x_0 + \text{span}\{p_{i-k-1},\ldots,p_{i-1}\}$.

GCR(k) is also proposed as a less expensive alternative to GCR. As in Orthomin(k), at most k direction vectors have to be saved. The cost per iteration is lower, since in general fewer than k direction vectors are used to compute p_{i+1}.

MR corresponds to the special case of k = 0 for both Orthomin(k) and GCR(k). It has very modest work and storage requirements, and in the symmetric case resembles the method of steepest descent.

In Table 2-1, we summarize the work and storage costs (excluding storage for A) of computing x_i for each of the methods. The storage for GCR includes space for the vectors, x_i, r_i, Ar_i, $p_0,\ldots,p_i$, and $Ap_0,\ldots,Ap_i$. Ap_i is computed recursively as

$$Ap_i = Ar_i + \sum_{j=\max(0,i-k)}^{i-1} b_j^{(i-1)} Ap_j \quad ,$$

so that the only matrix-vector product required is Ar_i. The entries for Orthomin(k) correspond to the requirements after the k'th iteration. The work for GCR(k) is the average over k+1 iterations.

GCR gives the exact solution to (2.1) in at most N iterations. The three variants do not in general display finite termination. In practice, however, all four methods tend to compute sufficiently accurate solutions in far fewer than N iterations.

We now present some error bounds for the four methods. Let $M := (A+A^T)/2$ denote the symmetric part of A, and let $R := -(A-A^T)/2$ denote the skew-symmetric part of A, so that $A = M - R$. Let $J := T^{-1}AT$ denote the Jordan canonical form of A. For any square matrix X, let $\sigma(X)$ denote the set of eigenvalues of X, let $\lambda_{min}(X)$ denote the eigenvalue of X of smallest absolute

value, let $\lambda_{max}(X)$ denote the eigenvalue of largest absolute value, and let $\rho(X)$ denote the spectral radius of X, $|\lambda_{max}(X)|$. If X is nonsingular, let $K(X)$ denote the condition number of X, defined to be $K(X) := \|X\|_2 \|X^{-1}\|_2$. Finally, let P_i denote the set of real polynomials q_i of degree less than or equal to i such that $q_i(0) = 1$.

The following bounds for GCR and GCR(k) are proved using (2.4):

Theorem 2.1: If $\{r_i\}$ is the sequence of residuals generated by GCR, then

$$\|r_i\|_2 \le \min_{q_i \in P_i} \|q_i(A)\|_2 \|r_0\|_2 \quad .$$

If A has a complete set of eigenvectors, then

$$\|r_i\|_2 \le K(T)\, M_i \|r_0\|_2 \quad ,$$

where

$$M_i := \min_{q_i \in P_i} \max_{\lambda \in \sigma(A)} |q_i(\lambda)| \quad .$$

Moreover, if A is normal, then

$$\|r_i\|_2 \le M_i \|r_0\|_2 \quad .$$

Theorem 2.2: If $\{r_i\}$ is the sequence of residuals generated by GCR(k), then

$$\|r_{j(k+1)}\|_2 \le [\min_{q_{k+1} \in P_{k+1}} \|q_{k+1}(A)\|_2]^j \|r_0\|_2 \quad .$$

If A has a complete set of eigenvectors, then

$$\|r_{j(k+1)}\|_2 \le (K(T))\, M_{k+1})^j \|r_0\|_2 \quad ,$$

and if A is normal, then

$$\|r_{j(k+1)}\|_2 \le (M_{k+1})^j \|r_0\|_2 .$$

Finally, the following result implies that Orthomin(k) converges, and provides another error bound for GCR, GCR(k), and MR.

Theorem 2.3: If $\{r_i\}$ is the sequence of residuals generated by GCR, Orthomin(k), GCR(k), or MR, then

$$\|r_i\|_2 \le \left[1 - \frac{\lambda_{min}(M)^2}{\lambda_{max}(A^TA)}\right]^{i/2} \|r_0\|_2 \quad .$$

and

$$\|r_i\|_2 \le \left[1 - \frac{\lambda_{min}(M)^2}{\lambda_{min}(M)\lambda_{max}(M) + \rho(R)^2}\right]^{i/2} \|r_0\|_2 \quad .$$

Proofs of these results can be found in [4].

3. Implementation with Preconditioning

The methods presented in the previous section can be accelerated by preconditioning techniques. Let Q be some nonsingular matrix. The solution to (2.1) can be found by solving any of the alternative problems

$$\tilde{A}\,\tilde{x} = [Q^{-1}A]\,[x] = [Q^{-1}f] = \tilde{f} \quad ; \tag{3.1}$$

$$\tilde{A}\,\tilde{x} = [A\,Q^{-1}]\,[Qx] = [f] = \tilde{f} \quad ; \tag{3.2}$$

$$\tilde{A}\,\tilde{x} = [Q_1^{-1}A\,Q_2^{-1}]\,[Q_2x] = [Q_1^{-1}f] = \tilde{f} \quad ; \tag{3.3}$$

where Q is (formally) factored into the product Q_1Q_2. If systems of equations having Q as coefficient matrix can be solved easily, then the use of Q as

preconditioning may greatly speed the convergence of GCR and its variants. In this section, we discuss the implementation of *preconditioned versions* of the four methods.

At each step, the approximate solutions generated by GCR et. al. minimize the Euclidean norm of the *residual over some subspace*. When preconditioning is used, the quantity minimized depends on the technique of applying the preconditioning. For example, if GCR is applied to (3.1), then $\|Q^{-1}(f-Ax_i)\|_2$ is minimized at each step. The residual of (3.2) is the same as the residual of the original problem (2.1). In this paper, we restrict our attention to this version of the preconditioned problem.

GCR can be implemented to solve (2.1) using (3.2) as follows:

<u>Algorithm 3.1</u>: The preconditioned generalized conjugate residual method:

Choose x_0 .

Compute $r_0 = f - Ax_0$.

Set $p_0 = r_0$.

FOR i = 0 STEP 1 UNTIL Convergence DO

$$a_i = \frac{(r_i, Ap_i)}{(Ap_i, Ap_i)}$$

$$x_{i+1} = x_i + a_i p_i$$

$$r_{i+1} = r_i - a_i A p_i$$

$$b_j^{(i)} = -\frac{(AQ^{-1}r_{i+1}, Ap_j)}{(Ap_j, Ap_j)} \quad , \quad j \le i$$

$$p_{i+1} = Q^{-1}r_{i+1} + \sum_{j=0}^{i} b_j^{(i)} p_j \quad .$$

The work per iteration for preconditioned GCR is identical to that for the unpreconditioned version, except that the matrix vector product is replaced by a preconditioned matrix-vector product $AQ^{-1}r_{i+1}$. In general, this operation is performed in two steps: a system of equations with coefficient matrix Q is solved for $Q^{-1}r_{i+1}$, and the result is multiplied by A. For some *preconditionings based on the incomplete factorization of A, more efficient* techniques for performing this operation have been developed [5].

In addition to the extra storage required for Q, preconditioned GCR *requires one more vector of storage than the unpreconditioned version, for* $Q^{-1}r_i$.

The implementations of Orthomin(k), GCR(k), and MR are analogous to Algorithm 3.1. *For all three methods, the work per iteration differs from the* unpreconditioned versions only in the cost of the matrix vector product. Again, extra storage is required for Q and $Q^{-1}r_i$.

4. Sample Problems

In this section, we describe two sample problems on which we tested the methods. Consider the elliptic differential equation

$$-(u_{xx} + u_{yy}) + \beta u_x = 0 \quad , \tag{4.1}$$

on the quarter plane *x > 0, y > 0, with boundary conditions*

$$u(x,0) = 0 \quad , \quad u(0,y) = 1 \quad , \tag{4.2}$$

$$u(x,y) \text{ bounded as } |x| + |y| \to \infty \quad .$$

For large β, the solution u has a boundary layer near y = 0, and is nearly equal to 1 elsewhere [7].

For the numerical solution of (4.1) and (4.2), we restrict the domain to the unit square (0,1) x (0,1), and impose the additional boundary conditions

$$u(x,1) = 1 \quad , \quad u_x(1,y) = 0 \quad . \tag{4.3}$$

The effect of the outflow boundary condition is to make the boundary layer in the numerical solution nonoscillatory [7]. The exact solution to (4.1) - (4.3) is not known. A two-dimensional representation of a numerical solution for $\beta = 100$ is shown in Figure 4-1.

For the first test problem, we discretize (4.1) using centered finite differences on a uniform n x n grid, with $h = \frac{1}{n+1}$ [17]. The right boundary condition is discretized by

$$u(x_{n+1},y_j) = u(x_n,y_j) \quad .$$

Let u_{ij} denote the approximation to $u(ih,jh)$. The difference equations for the discretized problem are then

$$4u_{ij} - u_{i,j-1} - (1 + \frac{\beta h}{2})u_{i-1,j} - (1 - \frac{\beta h}{2})u_{i+1,j}$$
$$- u_{i,j+1} = 0 \ , \qquad 1 \le j \le n \quad , \quad 1 \le i < n \quad , \tag{4.4}$$
$$(3 + \frac{\beta h}{2})u_{nj} - u_{n,j-1} - (1+\frac{\beta h}{2})u_{n-1,j} - u_{n,j+1} = 0 \ ,$$
$$1 \le j \le n \quad .$$

where

$$u_{ij} = \begin{cases} 0 & j = 0 \quad , \\ 1 & i = 0 \text{ or } j = 1 \quad . \end{cases} \tag{4.5}$$

If the unknowns u_{ij} are ordered in the natural row-by-row manner, then

(4.4) can be expressed as an N x N block tridiagonal system of linear equations

$$A\,x := \begin{bmatrix} T_1 & -W_1 & & \\ -V_2 & T_2 & \ddots & \\ & \ddots & \ddots & -W_{n-1} \\ & & -V_n & T_n \end{bmatrix} x = f \ , \tag{4.6}$$

where for $1 \le i \le n$, T_i is the n x n tridiagonal matrix

$$\begin{bmatrix} 4 & -(1-\beta h/2) & & & \\ -(1+\beta h/2) & 4 & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & -(1+\beta h/2) & 4 & -(1-\beta h/2) \\ & & & -(1+\beta h/2) & 3+\beta h/2 \end{bmatrix} \ .$$

and V_i and W_i are n x n identity matrices. The right hand side f is determined by (4.5), and $N = n^2$. A is a nonsymmetric matrix, and it has complex eigenvalues for $\frac{\beta h}{2} > 1$ [14].

For the second test problem, we resolve the boundary layer by introducing the change of coordinates

$$y(\eta) = y_0'\eta + (1 - y_0')\eta^4$$

[16], with $y_0' = \frac{1}{\beta}$. Letting $v(x,\eta) := u(x,y(\eta))$, equation (4.1) becomes

$$-\left[v_{xx} + \frac{1}{y'}(\frac{1}{y'}v_\eta)_\eta\right] + \beta v_x = 0$$

with boundary conditions

$$v(x,0) = 0 \quad , \qquad v(0,\eta) = 1 \quad ,$$
$$v(x,1) = 1 \quad , \qquad v_x(1,\eta) = 0 \quad .$$

Using a uniform $n \times n$ grid on the unit (x,η) square, with $h = \frac{1}{n+1}$, and $y'_j = y'(jh)$, the difference equations are

$$\begin{aligned}(2 + \frac{1}{y'_j}(\frac{1}{y'_{j+1/2}} + \frac{1}{y'_{j-1/2}}))v_{ij} &- \frac{1}{y'_j}\frac{1}{y'_{j-1/2}}v_{i,j-1} \\ - (1 + \frac{\beta h}{2})v_{i-1,j} &- (1 - \frac{\beta h}{2})v_{i+1,j} \\ - \frac{1}{y'_j}\frac{1}{y'_{j+1/2}}v_{i,j+1} &= 0 \ , \qquad 1 \le j \le n \ , \ 1 \le i < n \ . \qquad (4.7)\end{aligned}$$

$$\begin{aligned}(1 + \frac{1}{y'_j}(\frac{1}{y'_{j+1/2}} + \frac{1}{y'_{j-1/2}}) + \frac{\beta h}{2})v_{nj} &- \frac{1}{y'_j}\frac{1}{y'_{j-1/2}}v_{n,j-1} - (1 + \frac{\beta h}{2})v_{n-1,j} \\ - \frac{1}{y'_j}\frac{1}{y'_{j+1/2}}v_{n,j+1} &= 0 \ , \qquad 1 \le j \le n \ .\end{aligned}$$

The resulting *matrix equation* is also of block tridiagonal form (4.6), and nonsymmetry occurs in both sets of diagonals. Because y'_j is small for j near 0, the diagonal coefficients in T_j, V_j, and W_j are large for small j.

5. Numerical Experiments

In this section, we present the results of numerical experiments. We solved the discrete problems (4.4) and (4.7) with $h = \frac{1}{32}, \frac{1}{48}$, and $\frac{1}{64}$. The linear systems are of order $N = 961$, 2209, and 3969. Three values of β were used, $\beta = 10$, 100, and 1000. All computations were done in double precision on a DECSYSTEM-20.

We tested the algorithms MR, Orthomin(1), Orthomin(5), GCR(1), and GCR(5). We also used the nonsymmetric Chebyshev algorithm [10, 11], and the conjugate gradient method applied to the normal equations (CGN) to solve the

same set of problems. The cost per iteration of the Chebyshev algorithm is 2N multiplications plus one matrix-vector product. The overhead required by the Chebyshev algorithm for estimating eigenvalues is not included in the operation counts. The cost per iteration of CGN is 5N multiplications plus two matrix-vector products.

All of the methods were tested in conjunction with two preconditionings: the incomplete LU factorization (ILU) [12], and the modified incomplete LU factorization (MILU) [3, 6]. The preconditioned problems were formulated as in (3.2); this means that the variational quantity minimized by all the GCR-variants, as well by CGN, is the Euclidean norm of the residual of the original linear problem, $\|r_i\|_2$.

The ILU factorization is an approximate LU factorization of A into $Q_I = L_I U_I$ that satisfies

1. if $A_{ij} = 0$, then $[L_I]_{ij} = 0$ and $[U_I]_{ij} = 0$;
2. if $A_{ij} \neq 0$, then $[Q_I]_{ij} = A_{ij}$.

That is, the approximate factors are as sparse as the lower- and upper-triangular parts of A, respectively, and the product Q_I agrees with A in the *nonzero entries of A.*

The MILU factorization is an approximate LU factorization into $Q_M = L_M U_M$ that satisfies

1. if $A_{ij} = 0$, then $[L_M]_{ij} = 0$ and $[U_M]_{ij} = 0$;
2. if $A_{ij} \neq 0$, and $i \neq j$, then $[Q_M]_{ij} = A_{ij}$.

MILU differs from ILU in that the diagonal of Q_M is modified so that for $1 \le i \le N$,

$$\sum_{j=0}^{N} (A_{ij} - [Q_M]_{ij}) = 0 \quad .$$

The factors of Q_M are also as sparse as the lower- and upper triangular parts of A.

The preconditioned matrix vector products were implemented to take advantage of the two cyclic nature of the problems [5]. The cost of a preconditioned matrix-vector product is 9N multiplications. The stopping criterion for all the tests was

$$\frac{\|r_i\|_2}{\|r_0\|_2} < 10^{-6} .$$

Tables 5-1, 5-2, and 5-3 show the number of multiplications needed by each of the methods to satisfy the stopping criterion for Problem 1 (4.4). Figures 5-1, 5-2, and 5-3 graph the residual norm $\|r_i\|_2$ against the number of multiplications used by MR, Orthomin(1), GCR(1), CGN, and the Chebyshev method for Problem 1, with $h = \frac{1}{48}$ and ILU preconditioning. Figures 5-4, 5-5, and 5-6 graph $\|r_i\|_2$ against the number of multiplications for Problem 1 with $h=\frac{1}{48}$ and MILU preconditioning.

Tables 5-4, 5-5, and 5-6 show the number of multiplications needed by each of the methods to satisfy the stopping criterion for Problem 2 (4.7). Figures 5-7 through 5-12 graph residual norm against multiplications for Problem 2, with $h = \frac{1}{48}$.

REFERENCES

[1] R. Chandra, S. C. Eisenstat, and M. H. Schultz. Conjugate gradient methods for partial differential equations. In R. Vichnevetsky, Editor, Proceedings of the AICA International Symposium on Computer Methods for Partial Differential Equations, AICA, Lehigh University, Bethlehem, Pennsylvania, 1975, pp. 60-64.

[2] Rati Chandra. Conjugate Gradient Methods for Partial Differential Equations. Ph.D. Thesis, Department of Computer Science, Yale University, 1978. Also available as Research Report #129.

[3] Todd Dupont, Richard P. Kendall and H. H. Rachford Jr. An approximate factorization procedure for solving self-adjoint elliptic difference equations. SIAM Journal on Numerical Analysis 5:753-782, 1968.

[4] Stanley C. Eisenstat, Howard C. Elman, and Martin H. Schultz. Variational Iterative Methods for Nonsymmetric Systems of Linear Equations. 1981. To appear.

[5] Stanley C. Eisenstat. Efficient implementation of a class of conjugate gradient gradient methods. Technical Report 185, Yale University Department of Computer Science, August 1980.

[6] Ivar Gustafsson. A class of first order factorizations. BIT 18:142-156, 1978.

[7] G. W. Hedstrom and Albert Osterheld. The effect of cell Reynolds number on the computation of a boundary layer. Journal of Computational Physics 37:399-421, 1980.

[8] Magnus R. Hestenes and Eduard Stiefel. Methods of conjugate gradients for solving linear systems. Journal of Research of the National Bureau of Standards 49:409-435, 1952.

[9] David S. Kershaw. The incomplete Cholesky-conjugate gradient method for the iterative solution of systems of linear equations. Journal of Computational Physics 26:43-65, 1978.

[10] Thomas A. Manteuffel. The Tchebychev iteration for nonsymmetric linear systems. Numerische Mathematik 28:307-327, 1977.

[11] Thomas A. Manteuffel. Adaptive procedure for estimation of prameters for the nonsymmetric Tchebychev iteration. Numerische Mathematik 31:187-208, 1978.

[12] J. A. Meijerink and H. A. van der Vorst. An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix. Mathematics of Computation 31:148-162, 1977.

[13] M. Petravic and G. Kuo-Petravic. An ILUCG algorithm which minimizes in the Euclidean norm. Journal of Computational Physics 32:263-269, 1979.

[14] G. D. Smith. Numerical Solution of Partial Differential Equations: Finite Difference Methods. Oxford University Press, New York, 1978.

[15] Eduard L. Stiefel. Relaxationsmethoden bester strategie zur losung linearer gleichungssystems. Comment. Math. Helv. 29:157-179, 1955.

[16] John Strickwerder. Iterative methods for the numerical solution of second order elliptic equations with large first order terms. SIAM Journal on Scientific and Statistical Computing 1:119-130, 1980.

[17] Richard S. Varga. Matrix Iterative Analysis. Prentice Hall, New York, 1962.

[18] P. K. W. Vinsome. Orthomin, an iterative method for solving sparse sets of simultaneous linear equations. In Society of Petroleum Engineers of AIME, Proceedings of the Fourth Symposium on Reservoir Simulation. 1976, pp. 149-159.

Table 2-1 Work per iteration (mv denotes a matrix vector product) and storage requirements of GCR and variants.

	GCR	Orthomin(k)	GCR(k)	MR
Work/ Iteration	(3i+4)N + 1 mv	(3k+4)N + 1 mv	((3/2)k+4)N + 1 mv	4N + 1 mv
Storage	(2i+3)N	(2k+3)N	(2k+3)N	3N

Figure 4-1 Numerical solution to (4.1) - (4.3) for β = 100

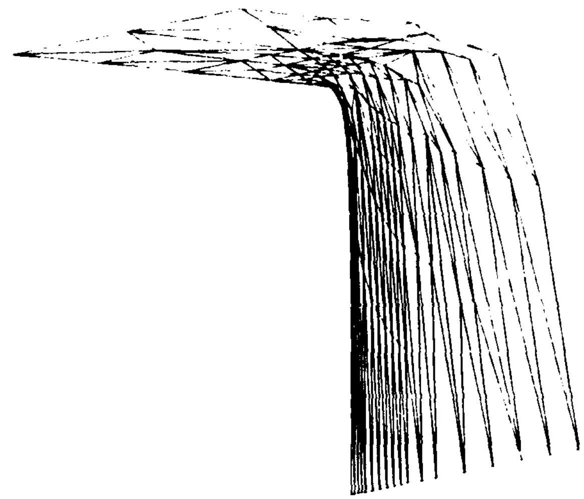


Table 5-1: Work required to reduce relative residual by factor 1.E 6.
Problem 1, Beta = 10

1/h	ILU 32	ILU 48	ILU 64	MILU 32	MILU 48	MILU 64
MR	1432409	7006113	>15000000	373921	1290477	3041573
Orthomin(1)	958893	3931397	10991957	275253	810701	1712141
Orthomin(5)	1057269	3236541	8369493	389069	1204533	2609973
GCR(1)	786989	3658793	11454393	279681	864793	1900381
GCR(5)	734829	2774717	6451161	329277	974477	2077701
CGN	1775376	8325204	>18000000	1014721	3752909	9840311
Chebyshev	476687	1390683	4835439	370915	977976	1849050

Table 5-2: Work required to reduce relative residual by factor 1.E 6.
Problem 1, Beta = 100

1/h	ILU 32	ILU 48	ILU 64	MILU 32	MILU 48	MILU 64
MR	238533	807065	1914733	275457	977681	2478153
Orthomin(1)	290445	1091213	2911573	336021	1196405	3227213
Orthomin(5)	469253	2312901	5600493	442525	1573989	4492893
GCR(1)	291989	991793	2420901	291989	963357	2243425
GCR(5)	446385	1929545	5641261	401581	1514549	3907113
CGN	666993	2597274	7576986	558328	1793354	4317798
Chebyshev	206677	646156	1597302	300142	1098061	2195298

Table 5-3: Work required to reduce relative residual by factor 1.E 6
Problem 1, Beta = 1000

1/h	ILU 32	ILU 48	ILU 64	MILU 32	MILU 48	MILU 64
MR	164685	465833	1043993	115453	408961	1043993
Orthomin(1)	184101	530189	1207117	138525	460061	1207117
Orthomin(5)	255429	650349	1613133	175245	588773	1613133
GCR(1)	169681	420293	985597	126989	420293	1048725
GCR(5)	212169	579365	1366653	149289	550929	1366653
CGN	340998	888944	2054473	319265	939189	2235539
Chebyshev	175522	430003	991368	113212	381969	904806

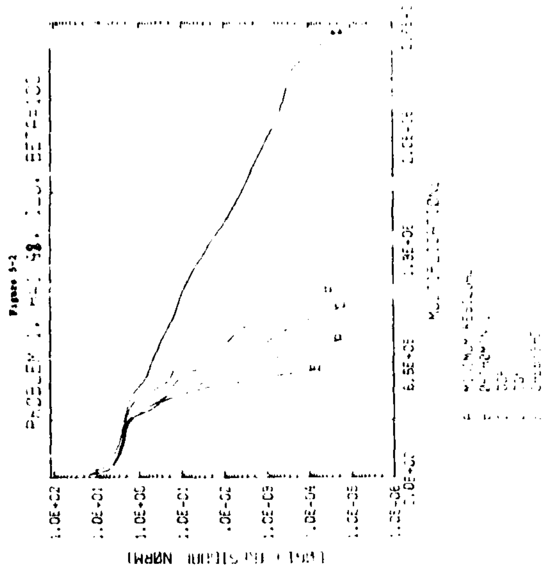


Figure 5-2

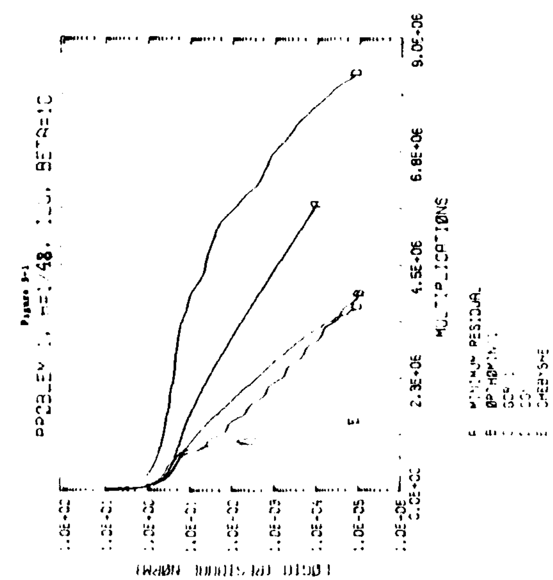


Figure 5-1

Figure 5-3

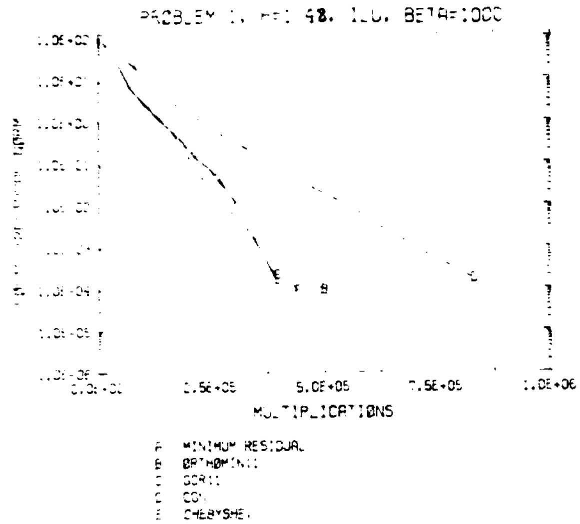


Figure 5-4

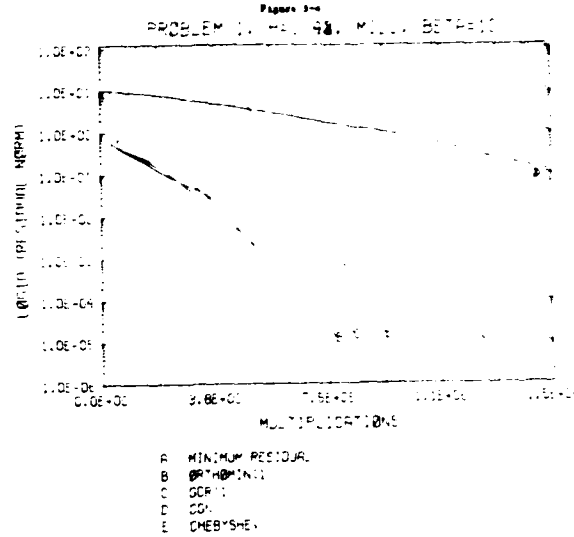


Figure 5-5

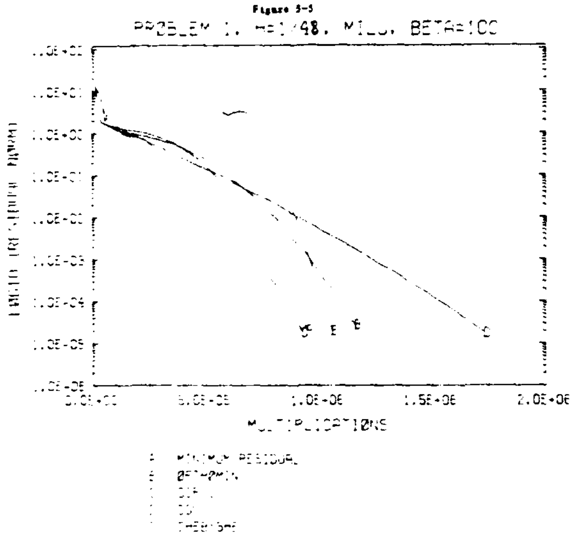


Figure 5-6

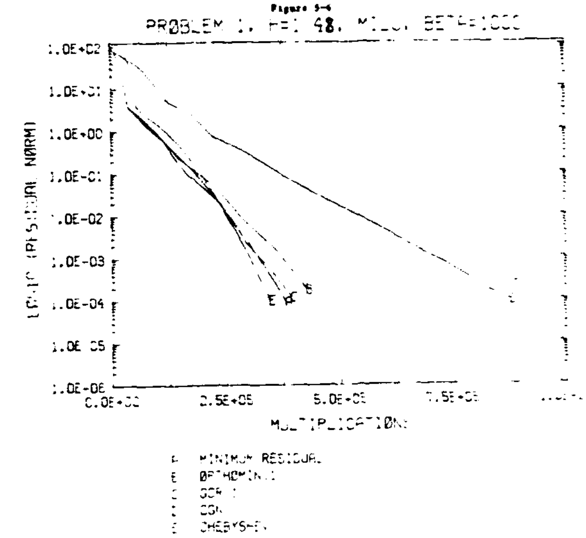


Table 5-4: Work required to reduce relative residual by factor 1.E-6.
Problem 2, Beta - 10

	ILU			MILU		
1/h	32	48	64	32	48	64
MR	1112401	5726493	>15000000	300073	949245	2170833
Orthomin(1)	746205	3089861	8719349	229677	635381	1333373
Orthomin(5)	977085	3174965	8369493	362341	1019805	2166933
GCR(1)	676989	3376357	10705177	236989	645857	1442989
GCR(5)	752905	2441182	5155321	284473	820965	1666797
CGN	1471114	7169569	18169347	775658	2647519	6581123
Chebyshev	518227	2106775	5008563	391685	929942	1805769

Table 5-5: Work required to reduce relative residual by factor 1.E-6.
Problem 2, Beta - 100

	ILU			MILU		
1/h	32	48	64	32	48	64
MR	300073	977681	2119613	287765	949245	2273273
Orthomin(1)	336021	1196405	3164085	381597	1301597	3164085
Orthomin(5)	683077	2436053	5822013	549437	1881869	4935933
GCR(1)	307181	1090357	2763945	334681	1090357	2472121
GCR(5)	539649	2006301	4582437	473113	1659509	4096497
CGN	601794	2295804	6762189	536595	1743109	4046199
Chebyshev	258602	766241	4316067	464380	1290197	2714670

Table 5-6: Work required to reduce relative residual by factor 1.E-6.
Problem 2, Beta - 1000

	ILU			MILU		
1/h	32	48	64	32	48	64
MR	373921	1176733	2939133	336997	1318913	4424513
Orthomin(1)	396789	1406789	3542853	411981	1722365	6131101
Orthomin(5)	576165	2251325	5489733	549437	2682357	7483413
GCR(1)	346989	1118793	2649597	346989	1344357	3678729
GCR(5)	500613	1929545	4021461	473113	2171145	5740113
CGN	601794	1893844	4498864	688726	2094824	4951529
Chebyshev	403992	1410282	3060918	424762	1482333	3450447

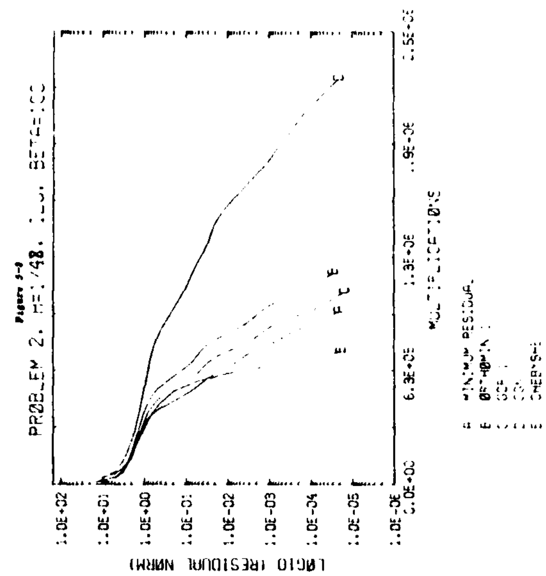


Figure 5-8

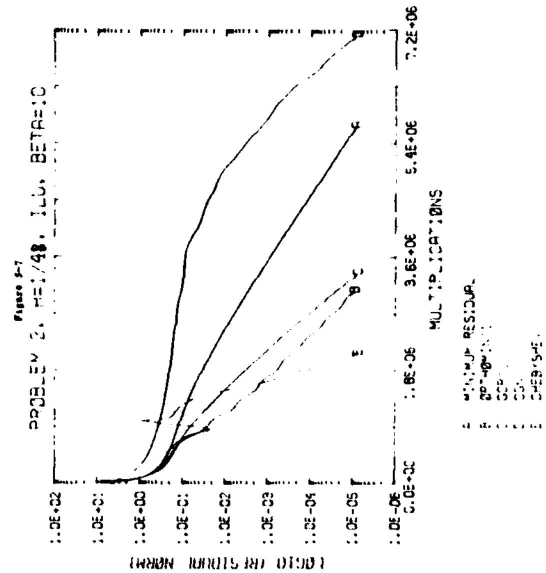


Figure 5-7

Figure 5-9

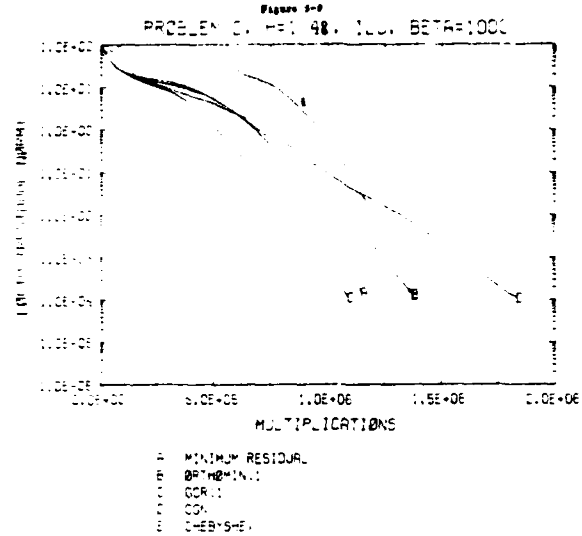


Figure 5-10

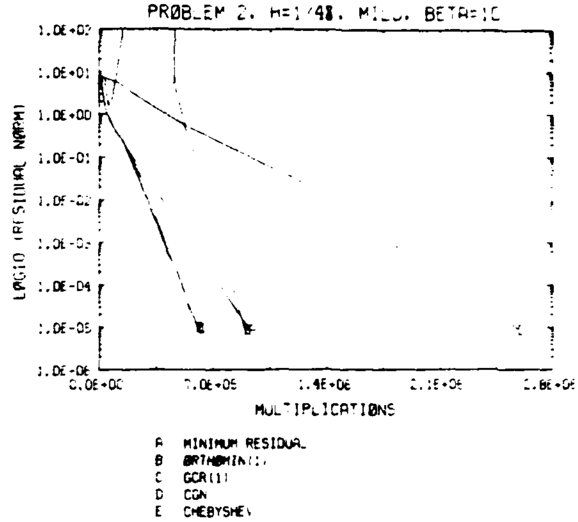


Figure 5-11

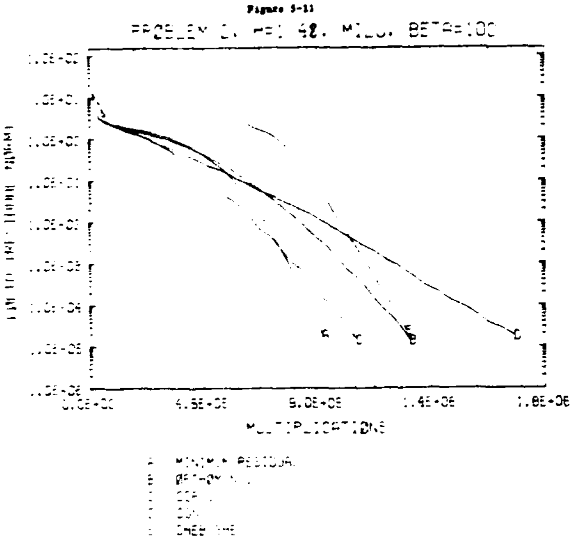
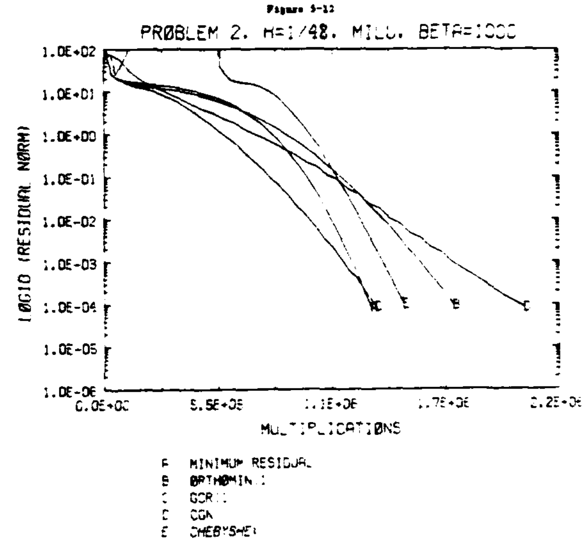


Figure 5-12



DAT
ILM